2010 International Conference on Intelligent Networking and Collaborative Systems

Using Bi-clustering Algorithm for Analyzing Online Users Activity in a Virtual Campus

Fatos Xhafa
Dept. of Languages and Informatics Systems
Technical University of Catalonia, Spain
EMail: fatos@lsi.upc.edu

Santi Caballé
Dept. of Computer Sci., Multimedia & Telecommunication
Open University of Catalonia, Spain
EMail: scaballe@uoc.edu

Leonard Barolli
Dept. of Information and Communication Engineering
Fukuoka Institute of Technology, Fukuoka, Japan
EMail: barolli@fit.ac.jp

Alberto Molina
Dept. of Languages and Informatics Systems
Technical University of Catalonia, Spain
EMail: amolina@lsi.upc.edu

Rozeta Miho
Faculty of Information Technologies
Technical University of Tirana, Albania
EMail: rmiho@fti.upt.al

***Abstract*—Data mining algorithms have been proved to be useful for the processing of large data sets in order to extract relevant information and knowledge. Such algorithms are also important for analyzing data collected from the users' activity users. One family of such data analysis is that of mining of log files of online applications that register the actions of online users during long periods of time. A relevant objective in this case is to study the behavior of online users and feedback the design processes of online applications to provide better usability and adaption to users' preferences. The context of this work is that of a virtual campus in which thousands of students and tutors carry out the learning and teaching activity using online applications. The information stored in log files of virtual campuses tend to be large, complex and heterogeneous in nature. Hence, their mining requires both efficient and intelligent processing and analysis of user interaction data during long-term learning activities. In this paper, we present a bi-clustering algorithm for processing large log data sets from the online daily activity of students in a real virtual campus. Our approach is useful to extract relevant knowledge about user activity such as navigation patterns, activities performed as well as to study time parameters related to such activities. The extracted information can be useful not only to students and tutors to stimulate and improve their experience when interacting with the system but also to the designers and developers of the virtual campus in order to better support the online teaching and learning.**

***Keywords*-Virtual Campus, Online Users, User Modelling, Mining Techniques, Bi-Clustering Algorithm**

I. Introduction and Motivation

Virtual campuses, virtual organizations and emerging virtual institutions [9] are new ways of organizing community-based activities by using IT technologies. These new paradigms of organization are sustained by the common interest of the members of the community to achieve their goals. Thus, in the case of virtual campuses, which is one of the most widely used form of virtual organizations in the today's teaching and learning activity, the members of the campus pursue academic goals. One of most distinguished characteristics of virtual organizations is *sharing*; members of the virtual organization share among them all sorts of resources, such as information, data, files and computational resources (computational power, data storages, etc.). Foster et al. [9] defined virtual organization as "*... a set of individuals and/or institutions defined by sharing rules form what we call a virtual organization (VO).*"

Virtual Campuses usually are supported by large or very large web sites which record not only accesses of their users but also all and each type of their activity interaction with the site during online sessions. As a consequence, there is a need to capture the users' interaction gathered in log files thus generating very large web log data files, which can be of a great variety of type and formats [24]. Therefore, there is a strong need for powerful solutions that record the large volume of interaction data and can be used to perform an efficient interaction analysis and knowledge extraction. To this end, data mining techniques can be used to find useful information and patterns on users' activity. For mining access log files of general purpose Web sites, many researchers have proposed different mining techniques [7], [10], [13].

The objective of mining log files of activity in virtual campuses is many-fold, amongst which we could distinguish: (a) User modelling and behavior; (b) Discovering access and browsing patterns; and, (c) Classification of user sessions.

978-0-7695-4278-2/10 $26.00 © 2010 IEEE
DOI 10.1109/INCOS.2010.15

User modeling [2], [3] is a mature research field mostly involved in the information technology context. User modeling is mainly utilized in software systems for inferring the users' goals, skills, knowledge, needs and preferences and thus achieving more adequate adaptation and personalization on the basis of the user activity pattern built [4]. This inference process relies in turn on being able to track the users' actions when interacting with the application such as the users' choice of buttons and menu items [11]. Therefore, on the one hand, the information captured from tracking is used by a user modeling algorithm in order to predict future users' actions, intentions and so on. On the other hand, based on the knowledge acquired from the user model, an adaptive system can adjust and personalize the system to individual user characteristics, preferences and needs. Indeed, adaptive systems [2] monitor the user model and automatically adjust the user interface and navigation or content provided by the system to accommodate such user differences as well as changes in user skills, knowledge and preferences. Thus, for instance, constantly maintaining the user model allows developers to receive continuous and useful feedback about the system's usability and adapt the user interface design to the actual users' needs whilst they evolve as time goes by. The ultimate aim of using user modeling and adaptive methods and techniques is to stimulate and improve the users' experience when interacting with the system [11].

The context of this paper are the Web-based applications that support on-line distance learning. These applications, due to the high degree of user interaction, take great advantage of the tracking-based techniques of user modeling, such as providing broader and better support for the users of Web-based educational systems [2]. Indeed, the data analysis of the information captured from the actions performed by learners is a core function for the modeling of the learner's behavior during the learning process and of the learning process itself as well. In addition, the building of learner models may help identify navigation patterns and adapt the system's usability to the actual learners' needs and thus stimulating the learning experience. In order to achieve the above goals, in this paper we present a bi-clustering algorithm [18] as a data mining technique for processing large log data sets from the online daily activity of students in a real virtual campus.

The rest of the paper is organized as follows. In Section II we present some of the related work on data mining techniques for supporting eLearning activity in Virtual Campuses, whose context as well as the importance and problems of modeling the students' behavior in Web-based environments is shown in Section III through the case of the Open University of Catalonia. The bi-clustering algorithm, which is used for mining the log data files of the online learning activity is presented in Section IV and the application architecture in Section V. We end the paper in Section VI with some conclusions and outline future work.

II. Use of Data Mining Techniques for Web Mining and Their Application to eLearning

The discovery of access patters from Web logs using mining techniques has been considered since at early stages of Web computing (e.g., [14]). Considerable research work has been done in mining various pattern information from Web logs to support improving the design and structure of a Web site and enabling organizations to function more efficiently [15], [17], [23]. Attention has been given to clustering users of Web sites into communities (e.g. [20]). This is particularly interesting for Virtual Campuses and Academic Web sites, where user profiling and particularization are important issues [21].

User navigation patterns are as well an interesting information to discover. Knowing patters of navigation can support designing Web sites that better support users to interact with the Web site [22]. For instance, in a Virtual Campus an online degree having different online rooms, it is interesting to know which resources the students access first and how they navigate during online sessions. This information would then feed back design processes for reorganizing Web sites based on user access patterns [10] or session categorization of users' activity [13].

Web usage mining attempts to discover useful knowledge from the secondary data obtained from the interactions of the users with the Web. Web usage mining has become very critical for effective Web site management, creating adaptive Web sites, business and support services, personalization, network traffic flow analysis and so on. Abraham [1] presented the important concepts of Web usage mining and its various practical applications.

Ciesielski and Lalani [6] presented the data mining of Web access logs from an academic Web site. Their goal was to use general purpose data mining algorithms to analyze visitors to the Web site and characterize them. As a result of their study, four different feature sets from the web logs were extracted. The authors used algorithms for classification, clustering, association finding and feature selection.

III. The Context of Virtual Campuses

Virtual Campuses are amongst the most important forms of virtual organizations, being the common goal the academic achievements of the members of the organization. Supporting all types and forms of online users (students, tutors, academic staff, visitors, etc.) is crucial for the functioning of Virtual Campuses. As a consequence, Virtual Campuses tend to be medium to large size and thus the efficient structuring of the Web sites and support to users is a major issue to achieve the Quality of Service (QoS) goals of the organization.

A. Our Virtual Campus

Our real web-based learning context is the Open University of Catalonia (UOC) [19], which offers distance

education through the Internet in different languages. As of this writing, about 54,000 students and 2,500 lecturers and tutors from everywhere participate in some of the about 30 official degrees and other PhD and post-graduate programs, resulting in more than 1,200 official courses. The campus is completely virtualized. It is made up of individual and community areas (e.g. personal electronic mailbox, virtual classrooms, digital library, on-line bars, virtual administration offices, etc.) through which users are continuously browsing in order to fully satisfy their learning, teaching, administrative and social needs.

From our experience at the UOC, the description and prediction of our students' behavior and navigation patterns when interacting with the campus is a first issue. Indeed, a well-designed system's usability is a key point to stimulate and satisfy the students' learning experience. In addition, the monitoring and evaluation of real, long-term, complex, problem-solving situations is a must in our context. Our goal is to understand and adapt the learning process and objects to the actual students' learning needs as well as to validate the campus' usability by monitoring the actual usage of the campus [4].

In order to achieve these goals, the analysis of the campus activity and specifically the users' traces captured while browsing the campus is essential in this context. The collection of this information in log files and the later analysis and interpretations of this information provide the means to model the user's behavior and activity patterns. For instance, from the log data it is possible to capture the different areas browsed by a student during his/her user session along with the timestamp when accessing to these areas. This allows us to know what the most popular areas are, how long in average students remain in each area, user session time in average and in different daily periods, navigation patterns combining both the most and the least visited areas, and so on and so forth.

However, in Web-based learning applications in general, extracting navigation and behavior patterns from the analysis of user interactions is a difficult task due to both the amount and the complexity of information generated. This makes its later treatment very tedious and time-consuming. Therefore, in order to construct a reliable, effective, useful learner models, this information has to enter a process to be effectively collected, processed and analyzed. During the first stage of this process, the most important issue while monitoring learning activity is the efficient collection and storage of the large amount of information generated. Given that such informational data may need a long time to be processed, Web-based learning systems have to be designed in a way that filter and pre-process the resulting information effectively. The aim is, on the one hand, to correctly collect and store the learning activity and, on the other hand, to increase the efficiency during the later data processing and analysis stages.

B. Difficulties in processing log data information of the Virtual Campus

The on-line Web-based campus of the UOC is made up of individual and community virtual areas such as mailbox, agenda, classrooms, library, secretary's office, and so on. Students and other users (lecturers, tutors, administrative staff, etc.) continuously browse these areas where they request for services to satisfy their particular needs and interests. For instance, students make strong use of email service so as to communicate with other students and lecturers as part of their learning process. As a result, the whole user interaction generates a huge amount of information in a day which is filtered and collected in large daily log files. Furthermore, this large information is found in an ill-structured highly redundant form as described next.

All users' requests are chiefly processed by a collection of Apache web servers as well as database servers and other secondary applications, all of which are providing service to the whole community and thus satisfying a large number of users. For load balance purposes, all HTTP traffic is smartly distributed among the different Apache web servers available and each web server stores in a log file each user request received and the information generated from processing it. Once a day (namely, at 01:00 a.m.), all web servers in a daily rotation merge their logs producing a single very large log file containing the whole user interaction with the campus performed in the last 24 hours.

A typical daily log file size may be up to 10 GB. This great amount of information is first pre-processed using filtering techniques in order to remove a lot of futile, non relevant information (e.g. information coming from automatic control processes, the uploading of graphical and format elements, etc.). However, after this pre-processing, about 1.8 GB of potentially useful information corresponding to 3,500,000 of log entries in average still remains [4].

Log file entries are structured following a type of format known as Common Log Format (CLF) [8] which is produced by most of web servers including Apache and is fairly configurable. For the purpose of registering the campus activity, log files entries were set up with the purpose of capturing the following information: *who* performed a request (i.e. user's IP address along with a session key that uniquely identifies a user session); *when* the request was processed (i.e. timestamp); *what* type of service was requested (a URL string format description of the server application providing the service requested along with the input values) and *where* (i.e. an absolute URL containing the full path to the server application providing the service requested).

At this point, we point out some problems arisen by dealing with these log files: (i)each explicit user request generates at least an entry in the log file and after being processed by a web server, other log entries are gener-

ated automatically from the response of this user request; (ii)certain non-trivial requests (e.g. user login) involve in turn requesting others and hence they may implicitly trigger new log entries; (iii)the *what* and *where* fields contain very similar information regarding the URL strings that describe the service requested and the parameters with the input values; (iv)certain information is found in a very primitive form and is represented as long text strings (e.g. user session key is 128-character string long). Therefore, there is a high degree of redundancy, tedious and ill-formatted information as well as incomplete as at some cases certain user actions do not generate any log entry (e.g. user may leave the campus by either closing or readdressing the browser) and thus these actions have to be inferred. As a consequence, treating this information is very costly in time and space, needing a great processing effort.

These drawbacks require a pre-processing of daily log data files obtained after merging those generated by the web servers as input so as to: (a) identify the log entries boundaries and extract the fields that make up each entry, (b) capture the specific information contained in the fields about users, time, sessions, areas, etc., (c) infer the missing information, (d) map the information obtained to typed data structures, and (e) store these data structures in a persistent support for later mining techniques.

IV. The Bi-clustering Algorithm

The log files of users' activity are used as a source to define the information of the composite event *"Who did what, when and where and how"*, where:

- *Who*: indicates the user of the action
- *What*: indicates the type of the action (which could be for instance, reading, downloading, creating new contents, modifying exiting contents, etc.)
- *When*: indicates the timestamp when the action occurred.
- *Where*: indicates the place in the Web site the action was performed (which could be a classroom, bulletin board, discussion forum, a particular space, etc. in the Web site).
- *How*: indicates the way the action was done.

This information fully defines the user action during online activity. It should be noted however that in most cases such information is not registered straightway in Web log files of user activities, rather, such information has to be extracted from information contained in the log file or log files of the Web site.

As can be seen, using the five parameters (who, what, when, where, how) the whole set of registers contained in the log file can be partitioned into different sets, which of them corresponding to a particular *feature* of users' activity. These feature sets are used as a basis for the bi-clustering algorithm, presented in this section. Essentially, feature sets provide the information to study the users' behavior at both particular and combined feature levels.

On the other hand, the classification of registers is done using *rules*, which use in input the values of the parameters (who, what, when, where, how) to classify the registers according to the considered features. In particular, we use the information of the parameter "how" to further distinguish registers belonging to the same feature set. For instance, we could distinguish two registers of the same action type but accomplished differently in two different parts/environments of the online system.

The registered modelling the event "who did what, when and where and how" are persistently stored in databases. It should as well be noted that different time intervals could be fixed for the study and thus only the registers belonging to specific time intervals would be used for mining purposes.

A. Bi-clustering: definition and notations

A matrix can be defined as a subset of elements that represent similar activity patterns with regard to a subset of features. Let A be a matrix denoted by $A = (X, Y)$ having rows $X = \{X_1, X_2, \dots X_n\}$ and columns $Y = \{Y_1, Y_2, \dots Y_m\}$. In this notation, we denote $A_{IJ} = (I, J)$ representing the sub-matrix of A that contains only the elements a_{ij} that belongs to the subsets of rows I and columns J.

Definition 1: A bi-cluster of elements of matrix A is an $A_{IJ} = (I, J)$ where $I = \{i_1, \dots, i_k\}$ is the subset of rows ($I \subseteq X$, $k \leq n$) and $J = \{j_1, \dots, j_s\}$ is the subset of rows ($J \subseteq Y$, $s \leq m$).

Bi-clusters can thus be seen as sub-matrices of a matrix representing features of elements. It should be noted that bi-clusters need not to be exclusive nor exhaustive.

Using the notation above, the bi-clustering problem can be defined as follows: given a matrix A, compute a family of bi-clusters $B_k = (I_k, J_k)$ so that each bi-cluster B_k satisfies certain "homogenous" properties leading to different types of *consistent bi-clustering*, including *bi-clustering admitting*, $\alpha-$*consistent bi-clustering*, $\beta-$*consistent bi-clustering*.

Similarly, as any clustering technique, the objective of bi-clustering is to compute/identify clusters of data grouped according to their similarities. In the case of bi-clustering the data is represented in a matrix in which rows are *features* of the elements of the sample under study represented by the columns of the matrix. For instance, in our case, the features characterize the actions of the users of the Virtual Campus. Thus, every element of the matrix represents how much the sample is expressed in the feature given by the row. More precisely, an element a_{uf} of the matrix represents how many registers corresponding to a user u satisfy a feature f.

The particularity of the bi-clustering is that partitioning is done in two dimensions of the matrix yielding to clustering according to elements and their features (in some sense this

can be seen as two inter-related partitions). We can thus define bi-clusters as follows.

Definition 2: A bi-cluster is a collection of pairs of subsets of elements and features $B = \{(S_1, F_1), (S_2, F_2), \ldots, (S_k, F_k)\}$, where k denotes the number of bi-clusters. Each bi-cluster (S_r, F_r) consists of two clusters: the S_r –the cluster of elements (samples) and F_r –the cluster of features.

B. The complexity of bi-clustering

The complexity of computing bi-clusters depends much on the way the bi-clustering problem is defined and how its quality is measured [18]. In most interesting formulations the bi-clustering problem is NP-hard problem. In fact, the problem is NP-hard even for the simple case when the matrix A is a binary matrix. In this case, from the matrix $A = (X, Y)$ we can construct a bi-partite graph $G = (L, R)$ in which a node $n_i \in L$ corresponds to a row of the matrix, a node $n_j \in R$ corresponds to a column of the matrix and a_{ij} is the weight of the edge (n_i, n_j). Any bi-cluster in the graph G corresponds to a bi-partite sub-graph and thus finding a bi-cluster of maximum size is equivalent of finding a sub-graph of maximum size in G, which is known to be NP-complete.

Besides the NP-hardness, the complexity of computing bi-clustering increases due to the information type to be analyzed. The information contains usually much noise due to the complexity of methods from which the information is captured. This requires careful pre-processing so that the resulting bi-clustering algorithm be robust to the noise contained in the data.

Finally, due to the large amount of data as well as to the large number of potential bi-clusters (especially when the number of features is large), the processing time can be a real issue. In fact, some bi-clusterings can be significantly overlapped or some others could be irrelevant. An efficient processing should thus take into account to extract only relevant bi-clusters and avoid overlapping among them.

C. Bi-clustering techniques

There has been proposed many bi-clustering techniques in the literature [16], including divide-and-conquer, iterative greedy, two-way, clustering, etc. Some of them aim to discover a set of bi-clusters (either of an *a priori* fixed cardinality or unknown cardinality). Other methods, such as the one by Cheng & Church, compute just one bi-cluster per execution.

The Cheng & Church algorithm: The Cheng & Church algorithm [5] uses the statistical measure of Mean Squared Residue (MSR). Let (I, J) be a bi-cluster, and e_{ij} an element of the matrix. We denote by e_{iJ} the mean value of the elements of row i and $j \in J$. Similarly, e_{Ij} denotes the mean value of the column j and $i \in I$. Finally, e_{IJ} denotes the mean value of all elements of the matrix with $i \in I$ and $j \in J$.

Definition 3: The residue R of an element e_{ij} of bi-cluster (I, J) is defined as $R(e_{ij}) = e_{ij} - e_{iJ} - e_{Ij} + e_{IJ}$. The Mean Squared Residue (MSR) of bi-cluster (I, J) is defined by:

$$H(I, J) = \frac{1}{|I||J|} \sum_{i \in I, j \in J} R^2(e_{ij}).$$

The MSR can be interpreted as a variance of elements of the bi-cluster and expresses the "coherence" of values of the matrix along rows and columns. It is interesting thus to find bi-clusters having small MSR values, which would mean larger coherence among the elements of the bi-cluster.

The algorithm (see Alg. 1) aims at finding a bi-cluster (sub-matrix) of largest possible size whose MSR is less than a given threshold δ. The algorithm takes in input the matrix and a threshold for the residue and works in two phases as follows.

In the first phase, the algorithm *eliminates* rows and columns of the matrix using residue values. Thus, for any row i, the algorithm uses the value of its residue $\frac{1}{|J|} \sum_{j \in J} R(e_{ij})$ is computed, and, for any column j, the value of its residue $\frac{1}{|I|} \sum_{i \in I} R(e_{ij})$. Then, the row or column of largest residue value is selected and eliminated from the sub-matrix. The computation process of the first phase ends when the residue of the sub-matrix $H(I, J) <= \delta$.

The second phase of the algorithm consists of *adding* rows and columns of the original matrix to the resulting sub-matrix of the first phase. The criteria for adding new rows and/or columns is that their residue should have small values. The second phase is finished when the threshold value for MSR is not surpassed.

One issue to take into account during the first phase is that due to the large size of the initial sub-matrix, deleting rows and columns can be computationally expensive. To alleviate this, *massive* deletion of rows and columns of large residue values can be done with the premise that their deletion will improve the overall MSR of the sub-matrix.

It should be noted from the above description that there exists a certain relationship between the size of the cluster and MSR value. Therefore, the threshold value for MSR will finally determine the size and quality of the obtained bi-cluster.

Implementation of Cheng & Church algorithm: In the implementation of the algorithm we have taken care to improve its performance as regards (a) no repeating of bi-clusters many times during the process and (b) efficient deleting and adding of rows and/or columns. Regarding the former, once a bi-cluster is computed, the rows and columns are extracted from the bi-cluster. As for the later, the operations of adding and deleting are implemented by marking rows and columns as "active" and "passive". Thus,

Algorithm 1 Cheng & Church Algorithm.

Input: Matrix E=(X,Y), threshold δ;
Output: Bi-cluster (I, J);
Initialization phase: $I = X$ and $J = Y$;
Deletion phase:
while $(MSR > \delta)$ **do**
    For any row, compute its MRS value as sum of MSR values of its elements;
    For any column, compute its MRS value as sum of MSR values of its elements;
    Compute the index of row or column of largest MRS value.
    Delete the resulting row from I or column from J.
end while
Addition phase:
while $(MSR <= \delta)$ **do**
    For any row, compute its MRS value
    For any column, compute its MRS value
    Compute the index of row or column of largest MRS value.
    Add the resulting row to I or column to J.
end while
return I and J;

only active rows are used during the computation process (at the beginning, all rows and columns are active). Finally, we have also paid attention to the efficient implementation of scoring function that computes the mean and MSR values of all (active) rows and columns.

The Cheng & Church algorithm is the main piece of the application for processing the log files of the Virtual Campus. We present in the next section the architecture of the application and technologies used for its implementation.

V. Application Architecture

The architecture of the application consists of several building blocks following standard software design patterns. The main building blocks include: (a) Rules, Conditions and Attributes Management; (b) Data mining; (c) Log file processing; (d) Course management; (e) Statistics management; (f) Diary user management; (g) Grid processing module; and, (h) Administration. Next, we briefly describe the most relevant part of this architecture for the sake of this paper.

A. Rules, conditions and attributes management

This module is devoted to the management of rules, conditions and attributes for the data mining module (see Fig. 1).

Rules Management: Rules are used to validate the registers in the log file against the features (see Fig. 2). By using the database of rules, validation of hits of feature(s) by registers of the log files is done.

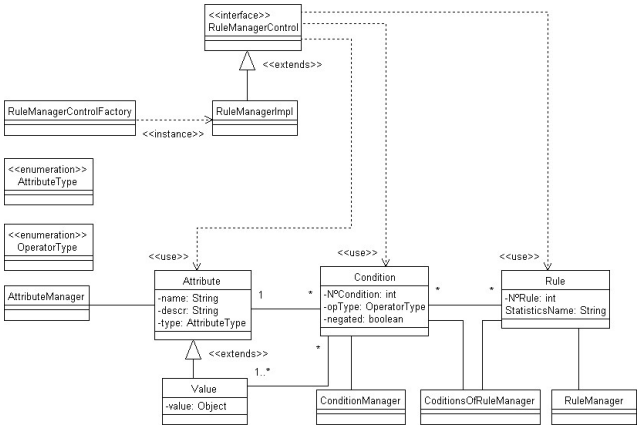

Figure 1. Rules, conditions and attributes model.

This package supports basic operations with rules such as listing existing rules, querying the list of rules and adding a new rule to the list.

For instance, to create a new rule enables to select a unique associated statistics from the list of statistics as well as selecting multiple conditions from the existing list of conditions.

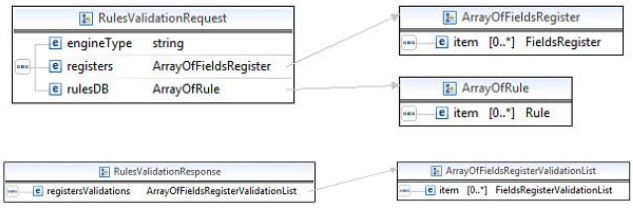

Figure 2. Rules validation request-response.

Conditions Management: Each rule is associated a set of conditions that define the premise of the rule (the *if* part of the rule). The relation between rules and conditions is multiple; each rule is associated one or more conditions and each condition can belong to zero or more rules (some conditions could be unassigned). Conditions are entities that model *filtering* on values of attributes in a line of the log file. Each condition corresponds to a unique attribute over which filtering is applied.

Attributes Management: The attributes are entities that model the concept of a token in a line of the log file. As such they can refer to more than one field of the log file lines and can have different formats. Each attribute is related to those conditions that apply over the possible values of the attribute in different lines of the log file.

B. Data mining

This module defines the functionalities of mining the log files and computing the bi-clusters of users of the Virtual

campus vs. the defined features (see Fig. 3).

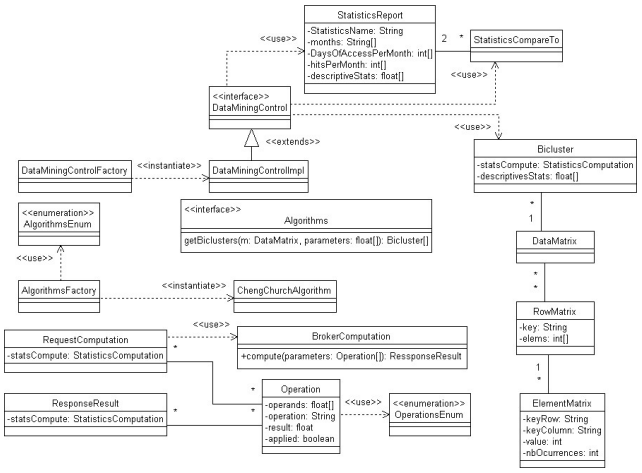

Figure 3. Data mining module.

C. Log files processing

The log files processing contains functionalities for processing log files generated by the online users of the Virtual Campus. Among these functionalities, there are the ones for formatting the log files (which originally are ill-structured and contain noise/irrelevant information) as well as the validation of the registers contained in the log files (see Fig. 4).

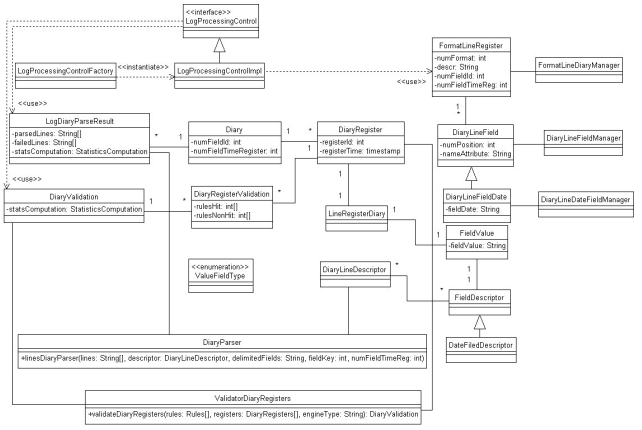

Figure 4. Log file processing module.

D. Grid processing module

One important issue addressed in the application architecture is the possibility to make a massive processing of the log files and speed up the processing and analysis of large and very large log files [24]. It can be easily seen that log data files of the Virtual Campus follow a Regularly Sequenced Data (RSD) pattern. In such format, log data is textual, record/register oriented and the boundaries between records are easily identifiable. The RSD format of log data is an important feature for its parallel processing. Indeed, since the boundaries between events/records are easily identifiable, processing log data files falls into the family of embarrassingly parallel applications.

Our design (see Fig. 5) enables thus to easily map log file processing capabilities to nodes of a large scale system, such as Grid systems, in which worker nodes can publish their processing capabilities using Grid services. The implementation is based on a Master-Worker paradigm [12] using Grid services for massive processing of log files.

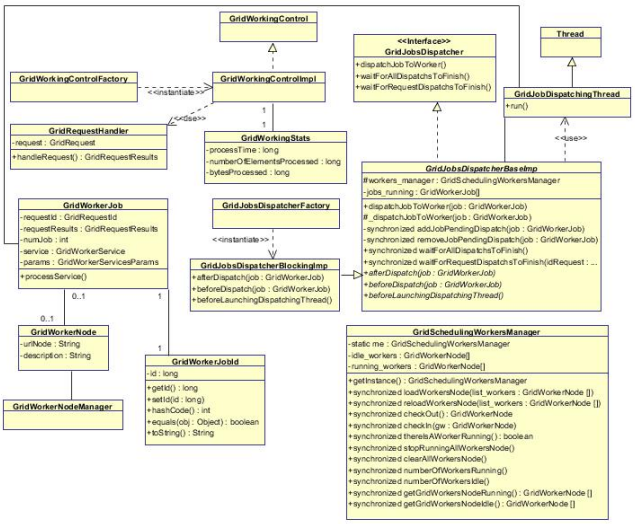

Figure 5. Grid processing module.

VI. Conclusions and Future Work

Web-based applications that support on-line distance education have been gaining a lot of attention due to the capability of offering training, long-life learning and education in general widely and easily available. In this context, it is essential to capture and understand the learner's behavior so as to predict future intentions, provide appropriate support and adapt the learning process and environment to learners' needs, preferences, knowledge, skills, and so on. The aim is to greatly stimulate the learning experience. To this end, we have proposed the use of a bi-clustering algorithm to extract useful information from the online activity of users of a virtual campus. The extracted information can be useful for many purposes, including: (a) better support to online learning and teaching activities in a Virtual Campus; (b) feed-back to designers and developers about the patterns of behavior of online users; and, (c) efficient distribution and use of resources of the Virtual Campuses in order to better match the need of online users. Our work showed that log data files can be used as a rich source of data

to accomplish these objectives. For the purposes of both showing the problem of dealing with log data and testing our prototype we have described and used the log data generated from the activity of online users of the Virtual Campus of the Open University of Catalonia.

At present, a prototype has been developed and tested. In our future work we will deploy the prototype in a Grid infrastructure and will fully evaluate both the performance of the application and the information extracted from log files as a decision support source.

ACKNOWLEDGMENT

This work has been partially supported by the European Commission under the Collaborative Project ALICE "Adaptive Learning via Intuitive/Interactive, Collaborative and Emotional System", VII Framework Programme, Theme ICT-2009.4.2 (Technology-Enhanced Learning), Grant Agreement n. 257639.

REFERENCES

[1] Abraham, A. (2003). Business Intelligence From Web Usage Mining. *Journal of Information and Knowledge Management (JIKM)*, Vol. 2, No. 4, 375-390.

[2] Brusilovsky, P. and Peylo, C. (2003) Adaptive and intelligent Web-based educational systems. *International Journal of Artificial Intelligence in Education*, 13(2-4), 159-172.

[3] Bushey, R., Mauney, JM., and Deelman, T. (1999) The Development of Behaviour-Based User Models for a Computer System. *Proc. of the Seventh International Conference on User Modeling (UM99)*, 109-118.

[4] Caballé, S., Xhafa, F., Fernndez, R., Daradoumis, Th. (2007). Efficient Enabling of Real Time User Modeling in On-line Campus. In *Proc. of the User Modeling 2007*, 365-369.

[5] Cheng Y., Church GM. (2000) Biclustering of Expression Data. *Proc. of the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB)*, 93-103.

[6] Ciesielski and Anand Lalani. (2003) Data Mining of Web Access Logs From an Academic Web Site. In *Proc. of the Third International Conference on Hybrid Intelligent Systems (HIS03)*, 1034-1043.

[7] Cooley, R., Mobasher, B., and Srivastava, J. (1999) Data preparation for mining world wide web browsing patterns. *Knowledge and Information Systems*, 1(1).

[8] Common Log Format: http://httpd.apache.org/docs/1.3/logs.html#common (as of August 2010).

[9] Foster, I., Kesselman, C. and Tuecke, S. (2001) The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of High-Performance Computing Applications*, 15(3), 200-222.

[10] Fu, Y., Creado, M., and Ju, C. (2001) Reorganizing websites based on user access patterns. *Proc. of the ACM CIKM International Conference on Information and Knowledge Management*, 583585.

[11] Gaudioso, E., Boticario, J.G. (2003) Towards web-based adaptive learning communities. *Proc. of Artificial Intelligence in Education*, 237-244.

[12] Goux, J., Kulkarni, S., Yoder, M., and Linderoth, J. (2000) An Enabling Framework for Master-Worker Applications on the Computational Grid. In *Proc. of the 9th IEEE international Symposium on High Performance Distributed Computing, 43*.

[13] Heer, J. and Huai, E. and Chi, H. (2002) Separating the swarm: categorization methods for user sessions on the web. *Proc. of the CHI 2002 Conference on Human Factors in Computing Systems (CHI-02)*, 243250.

[14] Joshi, K.P., Joshi, A., Yesha, Y., Krishnapuram, R. (1999) Warehousing and Mining Web Logs. *Proc. of the 2nd ACM CIKM Workshop on Web Information and Data Management*, pp. 63-68.

[15] Kitsuregawa, M., Toyoda, M., Pramudiono, I. (2002) Web Community Mining and Web Log Mining: Commodity Cluster Based Execution. *Proc. of the 13th Australasian Database Conference (ADC02)*, 3-10.

[16] Kriegel, H.-P., Krger, P., Zimek, A. (2009). Clustering High Dimensional Data: A Survey on Subspace Clustering, Pattern-based Clustering, and Correlation Clustering. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 3(1), 158.

[17] Masseglia, F., Poncelet, P., Cicchetti, R. (1999): An Efficient Algorithm for Web Usage Mining. *Networking and Information Systems Journal (NIS)*, 2(5-6), 571-603.

[18] Mirkin, B. (1996). Mathematical Classification and Clustering. Kluwer Academic Publishers. ISBN 0792341597.

[19] Open University of Catalonia: http://www.uoc.edu (as of August 2010).

[20] Paliouras, G., Papatheodorou, C., Karkaletsisi, V., Spyropoulous, C.D., (2000): Clustering the Users of Large Web Sites into Communities. *Proc. of the 17th International Conference on Machine Learning (ICML00)*, 719-726.

[21] Pazzani, M., Billsus, D. (1997): Learning and Revising User Profiles: The Identification of Interesting Web Sites. *Machine Learning*, 27, 313-331.

[22] Smith K.A. and Ng A. (2003), Web page clustering using a self-organizing map of user navigation patterns. *Decision Support Systems*, 35(2), 245-256.

[23] M. Spiliopoulou, M. and Pohle, C. and Faulstich, L.C. (2000) Improving the effectiveness of a website with web usage mining. *Proc. of the International Workshop on Web Usage Analysis and User Profiling*, 142-162.

[24] Xhafa, F., Paniagua, C., Barolli, L. and Caballé, S. (2010) A Parallel Grid-based Implementation for Real Time Processing of Event Log Data in Collaborative Applications. *Int. Journal of Web and Grid Services (IJWGS)*, volume 6, issue 2, 124-140.